An Architecture for Context-Aware Adaptive Data Stream Mining

Pari Delir Haghighi, Mohamed Medhat Gaber, Shonali Krishnaswamy, Arkady Zaslavsky, and Seng Loke

Center for Distributed Systems and Software Engineering
Monash University, Australia
{pari.delirhaghighi, shonali.krishnaswamy, Arkady Zaslavsky}@infotech.monash.edu.au
CSIRO ICT Center, Australia
Mohamed.gaber@csiro.au
Department of Computer Science and Computer Engineering
La Trobe University, Australia
s.loke@latrobe.edu.au

Abstract. In resource-constrained devices, adaptation of data stream processing to variations of data rates, availability of resources and environment changes is crucial for consistency and continuity of running applications. Context-aware adaptation, as a new dimension of research in data stream mining, enhances and optimizes distributed data stream processing tasks. Context-awareness is one of the key aspects of ubiquitous computing as applications' successful operations rely on detecting changes and adjusting accordingly. This paper presents a general architecture for context-aware adaptive mining of data streams that aims to dynamically and autonomously adjust data stream mining parameters according to changes in context and resource availability in distributed and heterogeneous computing environments.

1. Introduction

Processing of data streams due to their unpredictable and continuous nature [1, 2] is a challenging area of study. In the literature, various techniques and approaches have been presented to address the issues associated with data stream processing both in data mining and querying. However, recently the emergence and growth of mobile computing and networking and importance of using mobile devices for data stream mining in certain application domains (e.g. health or bushfire monitoring applications) have introduced new research challenges that need to be addressed. Data stream applications running on resource-constrained devices need not only to consider limitations of computational resources such as memory, battery level and CPU speed but also to take into account the issues of variable data rates, mobility, disconnections and environmental changes.

Nearly all pervasive systems utilize context to perform their tasks and this makes context-awareness an essential requirement of these systems [3, 4]. To perform data stream mining in heterogeneous and distributed computing environments, applications need to monitor context changes and react or adapt to them in order to maintain consistent and continuous operations.

Studying the current state-of-art in data stream mining [5-7] indicates that there are methods and algorithms introduced for efficiently mining high speed data streams in mobile devices such as Personal Digital Assistants (PDAs) but they have limited dynamic ability to adjust to a multitude of changing contextual parameters and have not been adequately equipped to cope with the distributed and heterogeneous nature of applications or the mobility of the users/devices that these techniques aim to support. One of the innovative adaptive works in data stream mining on resource-constrained devices is Algorithm Output Granularity (AOG) [8, 9] that provides adaptability with respect to the available memory on a device. Examples of light-weight data stream mining algorithms that have been developed using the AOG include LWC, LWClass and LWF [10].

In this paper we introduce a novel architecture for context-aware adaptive data stream mining that aims to provide real-time and dynamic strategies for adaptation and cost-efficiency by factoring in current context, availability of resources and distribution of resources and processing. This approach will significantly contribute to a range of application areas such as the mobile workforce, Intelligent Transportation Systems and sensor network applications. The summary of our main contributions are as follows:

- Incorporating context-awareness into data stream processing as a meta-level concept (i.e. situations) based on the Naïve Context Spaces (NCS) model;
- Enabling real-time and cost-efficient adaptation by matching context changes to a set of pre-defined application-specific situations and responding to changes accordingly;
- Introducing adaptation strategies with data stream processing parameters that are dynamically set/adjusted at run-time based on contextual changes, and shifting from purely reactive to proactive behavior;
- Ensuring the continuity and consistency of running operations on resource-constrained devices;

To explain different parts of our architecture throughout the course of the paper, we will use an example of a health monitoring application for heart patients. The details of the examples are only provided for the purpose of illustrating the points and may lack the necessary medical accuracy and correctness. Applications for healthcare biosensor networks are recently gaining popularity among people as they provide a convenient and safe way to monitor patients remotely and generate warnings and emergency calls. One of the main biosensors used for monitoring heart patients is ECG (Electrocardiogram) sensors that send heart beat rates as a continuous data stream to a PDA [11] or to a base station using an ISM band [12]. Data stream querying or mining needs to be performed on the ECG sensor streams locally on a mobile device or on a central workstation for monitoring and analyzing heart beats.

This paper is structured as follows: Section 2 provides a general view of our proposed architecture for context-aware adaptive data stream mining. Section 3 discusses context and situation modeling and how situations are inferred. Section 4 focuses on adaptation strategy and correlation functions. Finally section 5 concludes the paper and discusses future work.

2. An Architecture for Context-Aware Adaptive Data Stream Mining

In this section, we introduce an architecture for context-aware adaptive data stream mining. The architecture consists of two parts as shown in Figure 1. The first part is a situation manager that provides context-awareness and includes components for context modeling and inference. The second part, strategy manager, is responsible for adjusting adaptation strategy parameters based on correlation functions and invoking strategies.

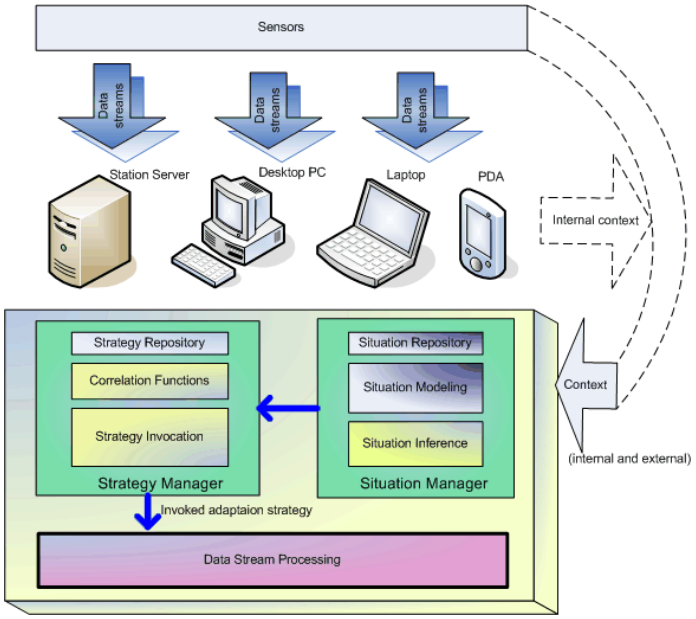

Figure 1. A general architecture for context-aware adaptive data stream mining

3. Situation Manager

The situation manager consists of three components: Situation repository, Situation Modeling and Situation Inference. These components work together to reason about the occurring situation based on the current context attribute values. Contextual information used for inferring situations may include sensed context collected from sensors, static context or internal context such as battery level of mobile devices.

3.1 Situation Modeling

We have based our context representation and modeling on the Naïve Context Spaces (NCS) Model [13, 14] but made changes to it to comply with our purpose. The NCS model and its extension in [15] are used as a powerful tool for reasoning about context and addressing uncertainties of sensed information. The core of the NCS model is the concept of situations. The NCS model represents contextual information as geometrical objects in multidimensional space called *situations* [13]. A *situation space* is a tuple of regions of attribute values related to a situation. Each *region* is a set of accepted values for an attribute based on a pre-defined predicate and each *context state* a collection of values of context attributes at the given time.

The NCS model extends the definition of context by describing it as "the set of facts, assumptions and predictions along with methods/algorithms of interpreting/ discovering/ processing that information" [15].

Using the NCS model, a situation occurs if every sensed context attribute value meets the predicate of the region set for the same type of attribute. We consider these situations as *known* situations. However, if the current context state does not match any of the pre-defined situations, it indicates the occurrence of an *unknown* situation. Any unknown situation can be *similar*/*dissimilar* to the situations already defined.

If we have an occurring situation S_i with a region of A_j, then for the sensed context attribute of a_i^t, we will have $a_i^t \in A_j$, and $A_{\min value} \le a_i^t \le A_{\max value}$ for continuous values. For any unknown situation there will be at least one context attribute value a_i^t that does not satisfy its associated region's predicate.

Considering our example for monitoring heart patients, from a list of related context attributes, we consider the following context attributes: temperature a_1 (0-50), age a_2 (20-120), location a_3 (HOME, NOT HOME), time a_4 (24 hours), heart rate a_5 (60-180), and Battery_level a_6 (0-100%). Weights are values from 0 to 1 that represent the importance of each context attribute in a situation and can have the total value of 1 per situation. Table 1 shows examples of pre-defined situations based on the aforementioned context attributes.

Table 1. Examples of situations

Situation	Context attributes	Regions and their predicates	Weight
S_1 sleep	a_1	<34	0.1
	a_2	<85	0.02
	a_3	HOME	0.03
	a_4	>10pm AND <7am	0.35
	a_5	<100	0.4
	a_6	>80%	0.1
S_2 Heat_stroke_ threat	a_1	>34	0.4
	a_2	>75	0.15
	a_3	NOT HOME	0.01
	a_4	>11am AND <8pm	0.01
	a_5	>100	0.4
	a_6	>40%	0.03
S_3 critical	a_1	>34	0.2
	a_2	>70	0.03
	a_3	NOT HOME	0.05
	a_4	>4pm AND <6pm (rush hour)	0.02
	a_5	>100	0.6
	a_6	<40%	0.1

3.2 Situation Repository

The situation repository contains a set of pre-defined situations that specify the most important regions of context attributes for the application however any major changes in context attribute values will be modeled as a situation (known or unknown) and used for adaptation of strategy parameters.

We have used XML schema (as shown in Figure 2) for defining our context model and XML documents for defining situations of different application domains.

We use JAXB to convert an XML document to java classes based on our XML schema. XML is a powerful and easy tool for the sharing of data across different applications. Applications can express their domain-related situations as an XML document in a simple way without requiring the knowledge of the underlying situation model.

```
<xsd:complexType name="Situations">
   <xsd:sequence>
      <xsd:element name="situation" minOccurs="1" maxOccurs="unbounded">
        <xsd:complexType>
        <xsd:sequence>
          <xsd:element name="situationName" type="xsd:string"/>
          <xsd:element name="significance" type="xsd:string"/>
          <xsd:element name="regions" type="Regions"/>

      ...

<xsd:complexType name="Regions">
  <xsd:sequence>
    <xsd:element name="region" minOccurs="1" maxOccurs="unbounded">
     <xsd:complexType>
     <xsd:sequence>
          <xsd:element name="attributeName" type="xsd:string"/>
          <xsd:element name="weight" type="xsd:string"/>
          <xsd:element name="predicates" type="Predicates"/>

      ...

<xsd:complexType name="Predicates">
  <xsd:sequence>
    <xsd:element name="predicate" minOccurs="1" maxOccurs="unbounded">
      <xsd:complexType>
        <xsd:sequence>
              <xsd:element name="condition" type="xsd:string"/>
                  <xsd:element name="valueType" type="xsd:string"/>
            <xsd:element name="values" type="Values"/>

      ...

<xsd:complexType name="Values">
  <xsd:sequence>
    <xsd:element name="attributevalue" minOccurs="1" maxOccurs="unbounded">
      <xsd:complexType>
```

Figure 2. XML schema of the situation model

3.3 Situation Inference

Situation inference is here referred to as discovering which pre-defined situation matches the current context state (collection of context attributes). Situation inference finds a perfect match for known situations and the closest match for unknown situations.

1. Let $R_{situations} = \{S_1, S_2, ..., S_n\}$ be the set of pre-defined situations in the *Situation Repository*.
2. Let context state $C = \{a_1^{\ t}, a_2^{\ t}, ..., a_m^{\ t}\}$ be the set of context attributes' values collected at time *t*.

3. We use the function *perfectMatch* $(C^t, R_{situations})$ to find a pre-defined situation that matches the current context state C_t. If there is not a perfect match, we find the most similar situation $S_{similar}$ using the State-Space difference Δ_{state}^{space} measures [15]. The function f calculates the distance between the context attribute and the region of accepted values, and $\hat{a}_i$ denotes the accepted region's absolute size.

$$\Delta_{state}^{space} = \sum_{i=1}^{n} w_i . f(a_i^s, a_i^r, \hat{a}_i) \cdot \quad (1)$$

4. After computing the state-space difference for all the pre-defined situations, the situation with the least difference value is selected as the most similar situation. The difference between a sensed context attribute and its closest value in the accepted region $(a_i^s - a_i^r)$ is later used for adjusting strategy parameters.

3.3.1 K-Nearest neighbor algorithm

To provide more accurate inference results we also perform the *k-nearest neighbor algorithm* to classify unknown situations based on the closest pre-defined situations. We first normalize instances and then measure Euclidean distance between the current context state and pre-defined situations to find the closet situation. We have tested the algorithm with k=1, k=3 and k=5, and the results are very similar. However, we intend to experiment this further with larger number of training examples and different application domains to determine the value of k.

4. Strategy Manager

To achieve real-time and dynamic adaptability, we use a set of parameterized adaptation strategies with their correlation functions. At run-time, the changes in context attribute values are used for adjusting parameter values in the corresponding strategy using its matching function. The strategy manager includes strategy repository, correlation functions and strategy invocation.

4.1 Strategy Repository

Every occurring situation may invoke a corresponding strategy to adjust the data stream mining parameters. We initially define a set of strategies with their parameters for each

pre-defined situation. Table 2 illustrates some examples of these strategies for our scenario.

Table 2. Examples of strategies

Strategies		parameters	Value	Situations
Str_1	Increase	window size (sec)	min= +5 max= +10 (non-linear)	S_1 Sleep
Str_2	Switch /change	process	lighter weight algorithm	
Str_3	Decrease	Window size (sec)	-6	S_2 heat_stroke
Str_4	Decrease	Window size (sec)	-12	S_3 Critical
Str_5	send warning/emergency call	-	-	

4.2 Strategy Invocation

Depending on which situation is currently occurring, a corresponding strategy is selected.

1. Let $R_{strategies} = \{Str_1, Str_2, ..., Str_n\}$ be the set of pre-defined strategies in the strategy repository.
2. The strategy invocation $SI(S_{occuring}, R_{repository}) = Str_i$ returns the adaptation strategy for the occurring situation.

Adaptation parameters need to be adjusted properly before they could be applied to data stream mining tasks. We use correlation functions to adjust the value of parameters based on context attribute values. For a data stream process P_i running on a data stream DS_i, the parameters that have been considered in resource-aware approaches [8, 16] include *Input rate, output rate, time frame, procesesing rate of data per unit, total time, available memory,* and *sampling method.* From the above parameters, those that can be adjusted by an adaptation strategy are: *Input rate, output rate, time frame,* and *sampling method*.

In addition to the above parameters, we consider the parameter of location L_i^{ds} for data streams, L_i^{device} for the mobile device D_i, and $L_i^{algorithm}$ for light weight algorithms that can be transferred from one location/device to another. The parameter of location addresses the aspect of mobility in devices and users as well as distribution of data streams and

algorithms on heterogeneous computing devices. Figure 3 illustrates adaptation of data stream processing.

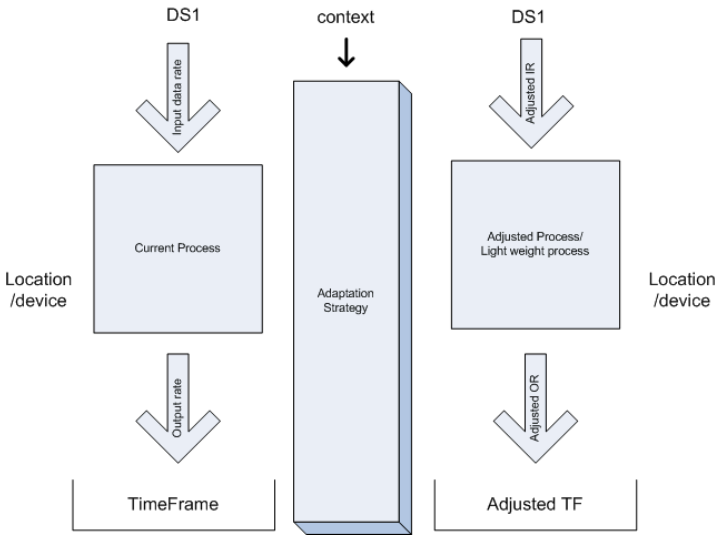

Figure 3. Adaptation of data stream processing

4.3 Correlation Functions

In order to make adjustments to strategy parameters, we use correlation functions that are pre-defined functions and vary from linear to non-linear. Correlation functions calculate strategy parameter values according to changes in context and the occurring situation. We envisage this concept by providing an example of context-aware adaptation from our scenario. In the heat-stroke-threat situation, the matching strategy reduces the window size from 30 sec (set value) to 24 sec (-6) as shown in Table 2. For a similar situation that meets all the accepted regions of the heat-stroke situation but its temperature value is 33 (refer to Table 1), we use a linear function (i.e. window size = $ax+b$ where $x=\Delta_{a_i^s}^{a_i^r}$ and a and b are set values) to adjust the parameter of window size. We discuss this further in the following steps:

1. Strategy invocation, $SI(S_{occuring}, R_{repository}) = Str_i$ returns the corresponding strategy Str_i for the current situation inferred by the situation manager.
2. Let $CF = \{f_1, f_2, ..., f_n\}$ be a set of pre-defined correlation functions.

3. Let $f_i(a_i, p_i) \rightarrow ap_i$ be a correlation function for Str_i that returns ap_i the adjusted value of the parameter p_i based on the relationship of the context attribute a_i and the parameter p_i where $p_i \in Str_i$ and $a_i \in S_{occuring}$.
4. The adjusted value ap_i of parameter p_i is then used in the strategy Str_i.

As an example of cost-efficient adaptation, we include a strategy for the sleep situation that uses a pre-defined non-linear function to increase the window size (i.e. reducing the reading rates of heart beat) based on the attribute of *time* during the night in order to save device resources. During the day (unknown situation), the window size is automatically adjusted according to the relationship of the time and window size in the sleep situation using a similar correlation function as shown in Figure 4. We consider sleep situation for patients who mainly suffer from heart irregularities under stress and tension that are more likely during the day.

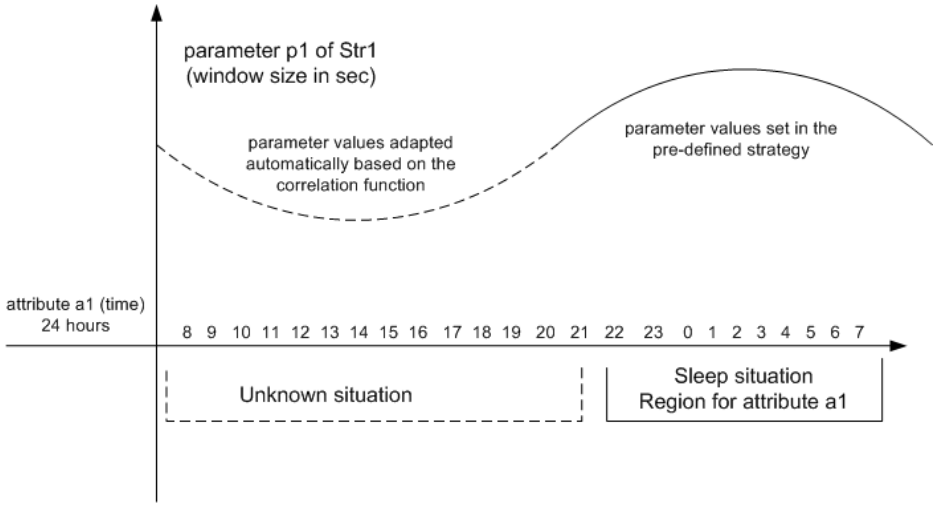

Figure 4. Using a quadratic function for adjusting parameters for sleep situation

5. Conclusion and Future Work

In distributed and heterogeneous environments, data stream applications have to cope with high data stream rates and limited computing resources such as memory or battery. Moreover, these applications need to address mobility, disconnections and environmental changes that are the norm in ubiquitous computing environments. Integrating data stream

processing with context-awareness enables applications to adjust to changes and continue their operations as expected.

In this paper we introduced a general architecture for context-aware adaptive data stream mining in heterogeneous computing environments. Our project is currently at the initial stages and our intention is to build a middleware based on the proposed architecture that will provide adaptation tasks and services on different data stream applications. While our current focus is on data stream mining, we see the potential for generalizing the approach for data stream processing including querying. Furthermore, there are still open issues in terms of distribution of data streams and mining algorithms that we intend to address in future work.

References

1. Babu, S., Widom, J.: Continuous Queries over Data Streams. ACM SIGMOD Record 30(3): pp. 109--120 (2001)
2. Golab, L., Ozsu, M.T.: Issues in Data Stream Management. SIGMOD Record 2003. 32(2): pp. 5--14 (2003)
3. Bunningen, A.H., Feng L. Apers, P.M.G.: Context for Ubiquitous Data Management. In: Proceedings of the 2005 International Workshop on Ubiquitous Data Management (UDM'05), pp. 17--28. IEEE Computer Society, Tokoyo (2005)
4. Davidyuk, O., Riekki, J., Rautio, V. Sun, J.: Context-Aware Middleware for Mobile Multimedia Applications. In: Proceedings of the 3rd International Conference on Mobile and Ubiquitous Multimedia. pp. 213--220. ACM Press, Maryland (2004)
5. Galan, M., Liu, H., Torkkola, K.: Intelligent Instance Selection of Data Streams for Smart Sensor Applications. SPIE Defense and Security Symposium, Intelligent Computing: Theory and Applications III., pp. 108-119. Orlando (2005)
6. Kargupta, H., Park, B, Pittie, S., Liu, L., Kushraj, D., pSarkar, K.: MobiMine: Monitoring the Stock Market from a PDA. SIGKDD Explorations, 3(2): pp. 37--46 (20020
7. Kargupta, H., Bhargava, R., Liu, K., Powers, M., Blair, P., Bushra, S., Dull, J., Sarkar, K., Klein, M., Vasa, M., Handy, D.: VEDAS: A Mobile and Distributed Data Stream Mining System for Real-Time Vehicle Monitoring. In: Proceedings of the SIAM International Data Mining Conference (SDM 2004), Orlando (2004)
8. Gaber, M.M., Krishnaswamy, Sh., and Zaslavsky, A., Resource-Aware Mining of Data Streams. Journal of Universal Computer Science. 11(8): pp. 1440--1453.
9. Gaber, M.M., Krishnaswamy, S., and Zaslavsky, A., On-board Mining of Data Streams in Sensor Networks, Advanced Methods of Knowledge Discovery from Complex Data, S. Badhyopadhyay, Maulik, U., Holder, L., and Cook,, D. (eds.), pp. 307--337. Springer, (2005)
10. Gaber, M.M., Krishnaswamy, Sh., and Zaslavsky, A. Cost-Efficient Mining Techniques for Data Streams. In: Proceedings of the 2nd Australasian Workshop on Data Mining and Web Intelligence (DMWI2004). Australian Computer Society, pp. 109--114. New Zealand (2004)
11. Brettlecker, G., Schuldt, H., and Schatz, R. Hyperdatabases for Peer-to-Peer Data Stream Processing. in Proceedings of the IEEE International Conference on Web Services (ICWS'04). pp. 358. IEEE Computer Society, San Diego (2004)

12. Chen, C., Agrawal, H., Cochinwala, M. and Rosenbluth, D. Stream query processing for healthcare bio-sensor applications. In: Proceedings of the 20th International Conference on Data Engineering (ICDE'04). pp. 791--794. IEEE Computer Society, Boston (2004)
13. Padovitz, A., Loke, S.W. and Zaslavsky, A. Towards a Theory of Context Spaces. In: Proceedings of the 2nd IEEE Annual Conference on Pervasive Computing and Communications, Workshop on Context Modeling and Reasoning (CoMoRea). IEEE Computer Society. Orlando (2004)
14. Padovitz, A., Loke, S.W., Zaslavsky, A. and Burg, B. . Towards a General Approach for Reasoning about Context, Situations and Uncertainty in Ubiquitous Sensing: Putting Geometrical Intuitions to Work. In: Proceedings of 2nd International Symposium on Ubiquitous Computing Systems (UCS'04). Japan (2004)
15. Padovitz, A., Context Management and Reasoning about Situations in Pervasive Computing, Caulfield School of Information Technology. Monash University: Australia. (2006)
16. Agarwal, I., Krishnaswamy, Sh., and Gaber M. M. Resource-Aware Ubiquitous Data Stream Querying In: Proceedings of the International Conference on Information and Automation. Sri Lanka (2005)